Korišćenje platna

Platno za rad se u Tkinter terminologiji naziva *kanvas objekat* (*canvas*). Dakle, kanvas objekat nam omogućava *crtanje* po prozoru aplikacije. *Crtanje* se u ovom kontekstu odnosi na linije, poligone, krugove (otvorene i zatvorene), slike, tekst, kao i grafiku. Pored ove funkcionalnosti, kanvas objekat omogućava dodavanje ugrađenih, kao i kreiranje sopstvenih, namenskih vidžeta. Objekti koji se dodaju na kanvas se dodaju kao stek, jedni na druge. Takođe, svi ovi objekti, kada se postave, dobijaju svoj identifikacioni broj tipa int, koji nam govori o tome gde se objekat nalazi na kanvas steku, kao i atribut tag – koji predstavlja ime pod kojim se taj objekat u kanvasu vodi (objekti koji se dodaju u kanvasu se mogu još nazivati i kanvas vidžeti). Pored nabrojanih mogućnosti, važno je napomenuti i da kanvas podržava i reagovanje na Tkinter događaje, kao i ostale kontrole. Iz svega ovoga, jasno je zašto se kanvas smatra jednom od najkorisnijih funkcionalnosti Tkintera.

Kanvas vidžeti su:

- arc (luk, odsečak) – metoda: `arc_method();`
- image (slik) – metoda: `create_image();`
- line (linija) – metoda: `create_line();`
- oval (elipsa) – metoda: `create_oval();`
- polygon (poligon) – metoda: `create_polygon();`
- rectangle (pravougaonik) – metoda: `create_rectangle();`
- window (dodavanje tkinter vidžeta u kanvas) – metoda: `create_window()`.

Za kreiranje ovih objekata uvek prvo moramo precizirati koordinate (x0, y0, x1, y1… xn, yn) koje se mogu proslediti pojedinačno, ali i kao lista ili n-torka. Takođe, svaka od ovih metoda ima povratnu vrednost – id, odnosno identifikacioni broj po kojem se objekat na kanvasu može locirati.

Crtanje linija

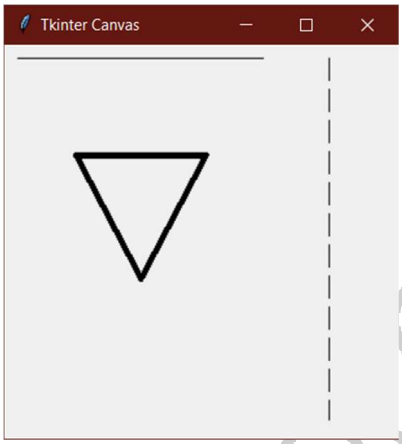
Linija je najjednostavniji oblik u geometriji. Linije na kanvasu možemo iscrtavati koristeći metodu `create_line()`. Za potrebe ovog, ali i ostalih primera, prvo moramo kreirati canvas objekat. Kako bismo olakšali dalji rad, dimenzije prilikom inicijalizacije kanvasa ćemo postaviti kao promenljive `canvas_width` i `canvas_height` – odnosno širina (x, horizontalna osa) i visina (y, vertikalna osa):

Primer crtanja linija

```
from tkinter import *
canvas_width = 300
canvas_height = 300
master = Tk()
master.title("Tkinter Canvas")
tk_canvas = Canvas(master,
           width=canvas_width,
           height=canvas_height)
```

```
tk_canvas.create_line(10, 20, 200, 10)
tk_canvas.create_line(250, 10, 250, 290, dash=(8, 4))
tk_canvas.create_line(55, 85, 155, 85, 105, 180, 55, 85, width = 5)
tk_canvas.pack(expand = YES, fill = BOTH)
master.mainloop()
```

Ako bismo pokrenuli ovaj primer, dobili bismo ovakav ekran:



Slika 6.1. Primer create_line metode

U prvoj metodi `tk_canvas.create_line(10, 20, 200, 10)` smo iscrtali jednostavnu horiziontalnu liniju kojoj smo zadali dve tačke: početnu i krajnju poziciju koje će linija spojiti. U ovom slučaju, koordinate su x0 = 10 i y0 = 20, x1 = 200 i y1 = 10.

Koordinate su ovde prosleđene pojedinačno, ali su isto tako mogle biti i u listi, na primer: `tk_canvas.create_line([10, 20, 200, 10])` ili u n-torkama, s tim što jedna koordinata čini jednu n-torku u kojoj se nalaze x i y komponente.

Drugi objekat smo na `tk_canvas` objekat smestili metodom `tk_canvas.create_line(250, 10, 250, 290, dash=(8, 4))`. U ovom slučaju smo iscrtali vertikalnu isprekidanu liniju. Isprekidanost linije smo postigli parametrom `dash` (crta, crtica), kojem prosleđujemo n-torku kao parametar sa dve vrednosti: prva vrednost predstavlja dužinu crtice, a druga vrednost predstavlja razmak između crtica.

U trećem slučaju, metodom `tk_canvas.create_line(55, 85, 155, 85, 105, 180, 55, 85, width = 5)` iscrtali smo više linija odjednom i time kreirali jednostavan jednakostranični trougao. Kod ovakvog pristupa crtanja trougla, bitno je voditi računa o početnim tačkama svake stranice, kako bi stranice bile međusobno spojene. U ovom slučaju je:

- prva tačka: x0 = 55, y0 = 85;
- druga tačka: x1 = 155, y1 = 85;
- treća tačka: x2 = 105, y2 = 180;
- četvrta tačka (potrebna radi spajanja poslednje linije): x3 = 55, y3 = 85.

Ove vrednosti dash argumenta su izražene u pikselima, baš kao i ostale vrednosti.

Izmeniti primer tako da se iscrtava paralelna linija sa linijom koja ima koordinate (200,200,50,50).

Kreiranje geometrijskih objekata

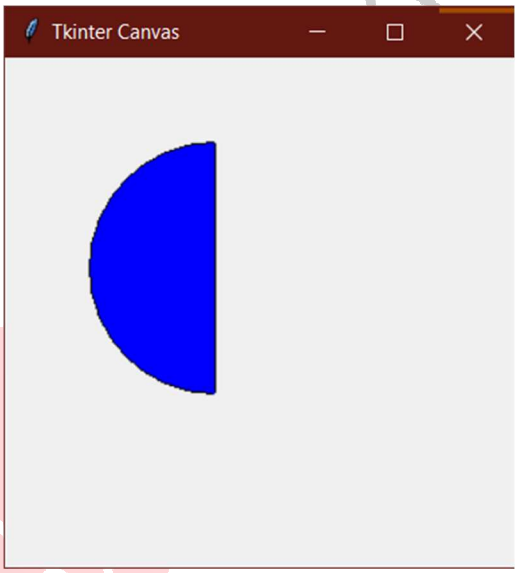
Geometrijski objekti koje Tkinter kanvas nativno podržava su: odsečci, elipse, poligoni i pravougaonici.

Odsečci (arcs)

Odsečke kreiramo metodom `create_arc()`, i to npr. sledećim pozivom:

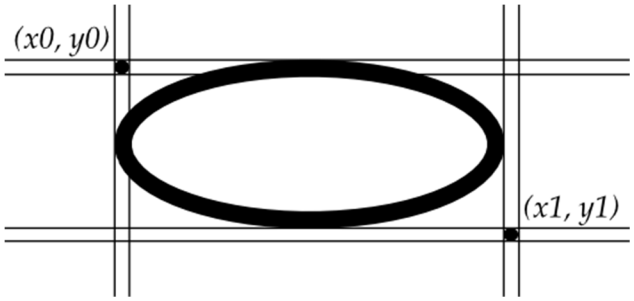
```
tk_canvas.create_arc(50, 50, 200, 200, start = 90, extent = 180,
fill="blue")
```

koji nam daje ovakav rezultat:



Slika 6.2. Primer korišćenja create_arc metode

Ovde se koordinate prosleđuju nešto drugačije. Ako zamislimo pravougaonik u kome će naš odsečak biti smešten – onda prosleđene koordinate predstavljaju suprotne uglove tog pravougaonika (gornji desni i donji levi ugao pravougaonika), pa nam je tako x0 = 50, y0 = 50 i x1 = 200, y1 = 200.

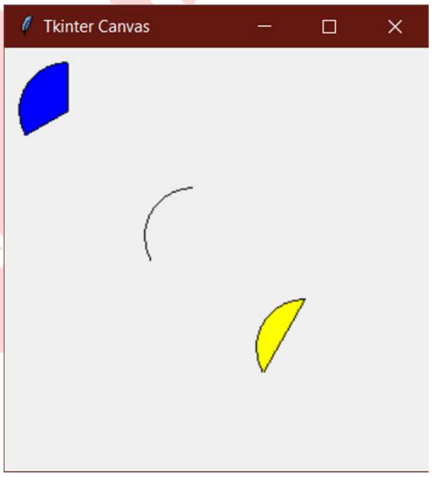


Slika 6.3. Princip prosleđivanja koordinata create_arc metodi[1]

Takođe, pored ovog specifičnog načina određivanja koordinata, ova metoda podržava i specifične argumente:

- Start – prosleđuje se vrednost tipa `int` koja pokazuje gde će isečak početi; ova vrednost se odnosi na horizontalnu, x osu i predstavlja, zapravo, na kom stepenu će taj odsečak početi; naravno, podržane vrednosti su od 0 do 360, pozitivne i negativne;
- Extent – prosleđuje se `int` vrednost koja označava širinu odsečka; opseg dozvoljenih vrednosti je takođe 0–360, sa pozitivnim i negativnim vrednostima; u našem primeru smo taj parametar postavili na vrednost 180, što nam je za rezultat dalo polovinu odsečka;
- Style – stil se odnosi na način na koji će odsečak biti iscrtan; podrazumevani argument je `PIESLICE`, dok su podržani i `CHORD` i `ARC`.

U radu sa `arc` objektima, takođe se često mogu koristiti parametri `outline` i `fill`, koji se odnose na boju ivice i boju unutar objekta.



Slika 6.4. Primeri korišćenja PIESLICE, CHORT i ARC stilova

Izmeniti primer tako da se iscrtava isečak u obliku Pakmenta(isečak od 0 do 270 stepeni).

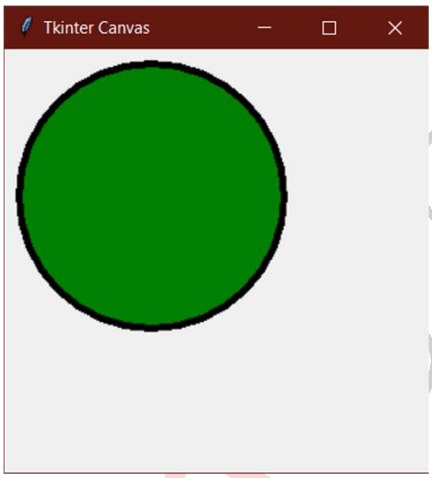
[1] https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/create_oval.html

Elipsa (oval)

Elipse u Tkinter kanvasu kreiramo preko `create_oval()` metode, koja se koristi slično kao i `create_arc()`, s tim što je jednostavnija od nje jer se ne koriste `start` i `extent` parametri, već se samo zadaju dve koordinate x i y ose, koje predstavljaju gornji desni i donji levi ugao zamišljenog pravougaonika koji će obuhvatati taj ovalni oblik. Takav poziv metode može izgledati ovako:

```
tk_canvas.create_oval(10, 10, 200, 200, outline="black", fill="green",
width = 5)
```

Ova linija nam daje sledeći rezultat:



Slika 6.5. Primer korišćenja create_oval metode

U ovom slučaju smo takođe prosledili početne koordinate gornje leve i donje desne ivice zamišljenog pravougaonika. Pošto su x i y komponente koordinata jednake jedna drugoj, dobili smo savršen krug. Da smo prosledili različite x i y komponente, dobili bismo ovalni oblik.

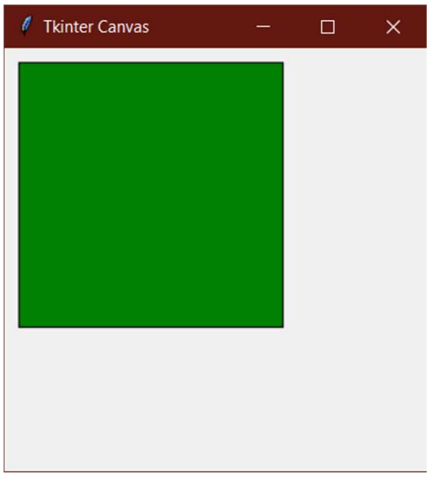
Dopunite primer sa crvenom elipsom koja treba da se nalazi unutar zelene elipse.

Pravougaonik (rectangle)

Pravougaonici se kreiraju slično kao i ovalni oblici, s tim što je ovde dovoljno samo proslediti koordinate gornje leve i donje desne tačke i pravougaonik će biti iscrtan:

```
tk_canvas.create_rectangle(10, 10, 200, 200, fill='green')
```

Pošto su x i y komponente jednake u koordinatama, dobili smo kvadrat:



Slika 6.6. Primer korišćenja create_rectangle metode

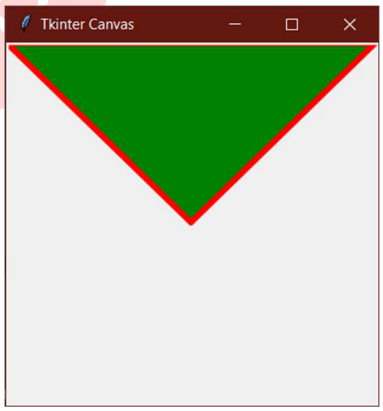
Dopunite primer sa još jednim kvadratom koji treba da se nalazi unutar zelenog kvadrata.

Poligoni (mnogouglovi) (polygon)

Rad sa poligonima u Tkinter kanvasu je sličan radu sa linijima. U radu sa linijama smo prosleđivali koordinate tačaka koje želimo da linija spoji. U slučaju poligona, odnosno mnogouglova, takođe prosleđujemo koordinate tačaka koje će linija spojiti, ali na kraju ne moramo sami izvršiti spajanje sa poslednjom, odnosno prvom tačkom kako bismo dobili ispunjen oblik, već ova metoda to radi umesto nas. Ova metoda je vrlo fleksibilna za korišćenje i omogućava nam jednostavno iscrtavanje dosta zahtevnijih oblika. Njeno korišćenje možemo pokazati na primeru jednakostraničnog trougla koristeći `canvas_width` i `canvas_height` promenljive koje smo definisali na početku:

```
points = (0,0,canvas_width/2, canvas_height/2, canvas_width, 0)
id_s = tk_canvas.create_polygon(points, fill = 'green', outline =
'red', width = 5)
```

Ovako izvršena `create_polygon()` metoda nam daje sledeći rezultat:



Slika 6.7. Primer korišćenja create_polygon metode

Koordinate su kreirane tako da se počinje od gornjeg levog ugla prozora (x0 = 0 ,y0 = 0); dalje se traži tačka koja je tačno na sredini preseka x i y ose prozora (x1 = canvas_height/2, y1 = canvas_width/2), dok je na kraju treća tačka u gornjoj desnoj ivici prozora aplikacije (x2 = canvas_width/2, y2 = canvas_height/2).

Kada se koristi ova metoda, potrebno je proslediti najmanje tri tačke.

Pitanje

Koju od ponuđenih metoda ćemo koristiti ako želimo da iscrtamo krug?

- create_line()
- **create_oval()**
- create_rectangle()

Objašnjenje:

Tačan odgovor je da ćemo za crtanje kruga (ili bilo kog drugog ovalnog oblika) koristiti metodu create_oval().

Izmenite kod tako da kvadrat bude prestavljen putem poligona.

Kanvas objekti i događaji

Već smo napomenuli da se kanvas objekti mogu povezivati sa događajima iz prethodne nastavne jedinice. To se čini posebnom kanvas metodom `tag_bind()`, gde prosleđujemo id kanvas objekta, odnosno povratnu vrednost neke od `create_..()` metoda, kao i tip događaja i funkciju koju je potrebno izvršiti. Recimo da želimo da dodamo još jedan pravougli trougao i omogućimo korisniku da ako klikne unutar jednog od ta dva poligona – obriše taj poligon sa kanvasa:

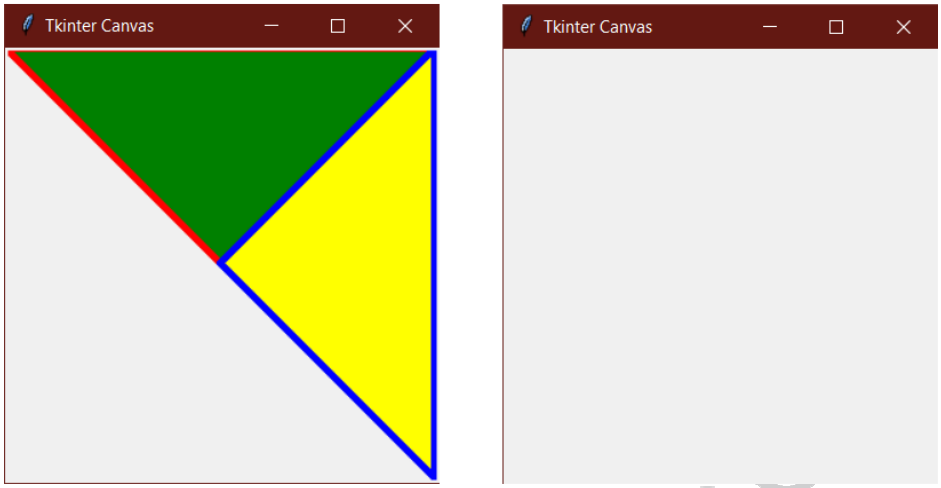
Primer događaja klik mišem nad kanvas objektom

```
points = (0,0,canvas_width/2, canvas_height/2, canvas_width, 0)
id_s = tk_canvas.create_polygon(points, fill = 'green', outline = 'red',
width = 5)

points = [canvas_height,canvas_height,canvas_height/2, canvas_width/2,
canvas_width, 0]
id_s_two = tk_canvas.create_polygon(points, fill = 'yellow', outline =
'blue', width = 5)

tk_canvas.tag_bind(id_s_two, '<Button-1>', lambda e:
tk_canvas.delete(id_s_two))
tk_canvas.tag_bind(id_s, '<Button-1>', lambda e: tk_canvas.delete(id_s))
```

Rezultat koda:



Slika 6.8. Pre i posle korisničke interakcije, odnosno klika na oba poligona

Na ovom primeru vidimo kako `tag_bind()` funkcioniše nad kanvas objektom. Takođe vidimo da je princip isti kada je reč o događajima: potreban je element nad kojim se događaj izvršava (prvi argument, povratna vrednost `create_...()` metoda), ime događaja koji će program osluškivati, kao i funkcija. U tom primeru smo iskoristili lambda funkciju, ali je moguće proslediti i klasično definisane funkcije. Brisanje se vrši jednom od više metoda kanvas objekta – `delete()`, kojoj prosleđujemo upravo id objekta na kanvasu. Ostale značajnije metode kanvas objekta su:

- `find_above(id)/find_below(id)` – metode kojima se prosleđuje id objekta, nakon čega vraćaju id objekta iznad/ispod tog objekta u steku koji sadrži objekte kanvasa;
- `find_overlapping(x0, y0, x1, y1)` – metoda koja vraća listu id-ja objekata koji dele najmanje jednu tačku datog pravougaonika;
- `move(id, x_amount, y_amount)` – metoda koja dati kanvas objekat pomera po x i y osi; koliko će se pomeriti, zavisi od `x_amount` i `y_amount`;
- `type(id)` – metoda koja vraća tip datog kanvas objekta (arc, oval, polygon, rectangle, text..)

Za ostatak metoda nad kanvas objektom pogledati zvaničnu dokumentaciju.

Rezime

- Platno za rad se u Tkinter terminologiji naziva *kanvas objekat* (*canvas*). Dakle, kanvas objekat nam omogućava *crtanje* po prozoru aplikacije.
- *Crtanje* se odnosi na linije, poligone, krugove (otvorene i zatvorene), slike, tekst, kao i grafiku.
- Kanvas objekat omogućava dodavanje ugrađenih, kao i kreiranje sopstvenih, namenskih vidžeta. Objekti koji se dodaju na kanvas se dodaju kao stek, jedni na druge.

- Linije predstavljaju najjednostavniji oblik u geometriji. Linije na kanvasu možemo iscrtavati create_line() metodom. Odsečke kreiramo metodom create_arc().
- Elipse u Tkinter kanvasu kreiramo preko create_oval() metode, koja se koristi slično metodi create_arc().
- Pravougaonici se u Tkinter kanvasu kreiraju pomoću metode create_rectangle().
- U slučaju poligona, odnosno mnogouglova, prosleđujemo koordinate tačaka koje će linija spojiti, ali na kraju ne moramo sami da izvršimo spajanje sa poslednjom, odnosno prvom tačkom kako bismo dobili ispunjen oblik, već ova metoda to radi umesto nas.